

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte Shawn Joseph Baranczyk and Brian Robert Muras

Appeal No. _____
Application No. 10/671,343

APPEAL BRIEF

Attorney Docket No. ROC920030213US1
Confirmation No. 7069

PATENT

CERTIFICATE OF ELECTRONIC TRANSMISSION

I hereby certify that this correspondence for Application No. 10/671,343 is being electronically transmitted to Technology Center 2132 via EFS-WEB, on February 6, 2008.

/Scott A. Stinebruner/
Scott A. Stinebruner, Reg. No. 38,323

February 6, 2008
Date

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

| | | | |
|------------------|--|-----------|--------------------|
| Applicant: | Shawn Baranczyk et al. | Art Unit: | 2132 |
| Application No.: | 10/671,343 | Examiner: | Farid Homavounmehr |
| Filed: | September 25, 2003 | | |
| For: | ENCRIPTION OF QUERY EXECUTION DETAILS IN A DATABASE MANAGEMENT SYSTEM | | |

Mail Stop Appeal Brief - Patents
Commissioner for Patent
P.O. Box 1450
Alexandria, VA 22213-1450

APPEAL BRIEF

I. REAL PARTY IN INTEREST

This application is assigned to International Business Machines Corporation, of Armonk, New York.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

Claims 1-11, 13-24 and 26-29 are pending, stand rejected, and are now on appeal.

IV. STATUS OF AMENDMENTS

An Amendment After Final was submitted on December 6, 2007, and accompanied the Notice of Appeal. Claim 29 was amended in an attempt to address the §101 rejection. In an Advisory Action dated January 22, 2008, the Examiner entered the amendment, but indicated that the amended claim did not overcome the §101 rejection, and as such, the §101 rejection of claim 29 is still maintained. Claim 29 is reproduced below in the Claims Appendix in its amended form.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Applicant's invention is generally directed to a database monitor that logs execution of a query in a database management system with one or more execution details encrypted or otherwise scrambled to restrict access to those execution details by unauthorized parties. (Application, page 4, lines 2-5). As a result, execution details such as the actual query statements, and values passed to parameter markers and/or host variables associated with the queries executed on a database management system may be kept confidential and accessible only to authorized parties having the ability to decrypt the execution details. (Application, page 4, lines 5-9).

A database monitor is used in many database management systems (DBMS's) to generate an execution log that records each query, as well as a number of execution details about the execution of that query, e.g., the type of access plan used to execute the query, the time and amount of resources used to execute the query, what indexes and other available resources were used during execution of the query, the number of records returned as a result of the query, etc. (Application, page 2, lines 9-15). Additional execution details that may also be logged include the actual query statement, as well as host variables and parameter markers used or incorporated in the query statement. A host variable is typically a variable defined in a host language, or that is generated by an SQL precompiler, and represents data to be inserted into the query during execution. A parameter marker, which is similar in function to a host variable, is typically used in dynamic SQL statements to represent positions in an SQL statement where data is to be inserted by an application. (Application, page 3, lines 1-7).

A database monitor execution log can be used to identify sub-optimal system settings to permit those settings to be adjusted to improve performance, as well as to identify the potential for other performance gains, e.g., to identify particular tables that are accessed with enough frequency to warrant the creation of additional indexes. (Application, page 2, lines 18-22). A database monitor execution log can also be used to optimize particular queries, e.g., to identify how to alter an original query to improve its execution on the database management system. (Application, page 2, lines 23-27).

Conventional database monitors have been found to suffer from the drawback that the contents of an execution log are stored in an unencrypted format. Some execution details, however, may include confidential and/or proprietary information that a party may not wish to be accessible to others. For example, some execution details such as the values passed to parameter markers or host variables during execution of a query may include confidential information such as employee social security numbers, salary information, etc. (Application, page 3, lines 10-15). Also, the query statements themselves may be confidential to an application developer, as many application developers expend substantial resources to develop complex SQL statements that perform complex operations in a highly tuned manner, and the ability of customers or competitors to view those statements when stored in an execution log could result in the loss of that valuable intellectual asset. (Application, page 3, lines 16-22).

Embodiments consistent with the invention address this problem associated with conventional database monitors by encrypting one or more details in an execution log to restrict access to such execution details by unauthorized parties. (Application, page 4, lines 2-5). The execution of a query is logged in a database management system by generating an encrypted representation of an execution detail for a query executed by the database management system (Application, page 11, lines 23-25; Fig. 3, block 74) and logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log (Application, page 11, lines 25-27; Fig. 3, block 70). In addition, the execution detail is decrypted in association with displaying the execution log (Application, page 12, lines 14-17; Fig. 4, blocks 86, 88).

For the convenience of the Board, each of the independent claims has been reproduced below and annotated with references to the specification and drawings to satisfy the requirement to concisely explain the claimed subject matter:

1. A method of executing a query (Application, page 11, lines 7-8) in a database management system (Application, page 8, lines 20-21; Fig. 1, block 28), the method comprising:

receiving an SQL statement from an application program coupled to the database management system (Application, page 11, lines 7-9; Fig. 3, block 60);

executing the SQL statement (Application, page 11, lines 12-14; Fig. 3, block 62);

encrypting the SQL statement to generate an encrypted representation of the SQL statement (Application, page 6, lines 5-7, page 11, lines 23-25; Fig. 3, block 74);

logging execution of the SQL statement in a database monitor by storing the encrypted representation of the SQL statement in an execution log managed by the database monitor (Application, page 11, lines 25-27; Fig. 3, block 70);

displaying the execution log, including retrieving the encrypted representation of the SQL statement from the execution log (Application, page 12, lines 3-5; Fig. 4, block 82), decrypting the encrypted representation of the SQL statement to generate an unencrypted representation of the SQL statement (Application, page 12, lines 14-17; Fig. 4, block 88), and displaying the unencrypted representation of the SQL statement (Application, page 12, lines 12-17; Fig. 4, block 86).

3. A method of logging query execution (Application, page 6, lines 2-5) in a database management system (Application, page 8, lines 20-21; Fig. 1, block 28), the method comprising:

generating an encrypted representation of an execution detail for a query executed by the database management system (Application, page 11, lines 23-25; Fig. 3, block 74);

logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log (Application, page 11, lines 25-27; Fig. 3, block 70); and

decrypting the execution detail in association with displaying the execution log (Application, page 12, lines 14-17; Fig. 4, blocks 86, 88).

16. An apparatus (Application, page 7, lines 4-6; Fig. 1, block 10), comprising:
at least one processor (Application, page 7, lines 14-15; Fig. 1, block 12);
a memory (Application, page 7, lines 15-16; Fig. 1, block 14) within which is stored an execution log (Application, page 10, lines 19-22; Fig. 2, block 54); and

program code configured to be executed by the at least one processor to log query execution (Application, page 6, lines 2-5) in a database management system (Application, page 8, lines 20-21; Fig. 1, block 28) by generating an encrypted representation of an execution detail for a query executed by the database management system (Application, page 11, lines 23-25; Fig. 3, block 74), logging the execution detail for the query in the execution log by storing the encrypted representation thereof in the execution log (Application, page 11, lines 25-27; Fig. 3, block 70), and decrypting the execution detail in association with displaying the execution log (Application, page 12, lines 14-17; Fig. 4, blocks 86, 88).

29. A program product (Application, page 9, lines 3-20), comprising:

program code (Application, page 9, lines 3-20) configured upon execution to log query execution (Application, page 6, lines 2-5) in a database management system (Application, page 8, lines 20-21; Fig. 1, block 28) by generating an encrypted representation of an execution detail for a query executed by the

database management system (Application, page 11, lines 23-25; Fig. 3, block 74), and logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log; and

a recordable computer readable medium (Application, page 9, lines 3-20) storing the program code.

Other support for the claimed subject matter may generally be found in Figs. 3 and 4 and the accompanying text at pages 11-12 of the Application as filed. In addition, it should be noted that, as none of the claims recite any means plus function or step plus function elements, no identification of such elements is required pursuant to 37 CFR §41.37(c)(1)(v). Furthermore, there is no requirement in 37 CFR §41.37(c)(1)(v) to provide support for the subject matter in the separately argued dependent claims, as none of these claims recite means plus function or step plus function elements, and so no discussion of any of these claims is provided.

VI. GROUND S OF REJECTION TO BE REVIEWED ON APPEAL

- A. Claim 29 is rejected under 35 U.S.C. §101 as being directed to non-statutory subject matter.
- B. Claims 1-11, 13-24 and 26-29 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,713,018 to Chan (hereinafter “Chan”).

VII. ARGUMENT

Applicant respectfully submits that the Examiner’s rejections of claims 1-11, 13-24 and 26-29 are not supported on the record, and should be reversed. Applicant will hereinafter address the Examiner’s rejections in the general order presented in the Final Office Action. Within the discussion of each rejection, the various claims that are the subject of the Examiner’s rejections will further be addressed in order, starting with the independent claims, and then

addressing various dependent claims reciting additional subject matter that is distinguishable from the prior art of record. In some cases, specific discussions of particular claims are not made in the interests of streamlining the appeal. The omission of a discussion with respect to any particular claim, however, should not be interpreted as an acquiescence as to the merits of the Examiner's rejection of the claim, particularly with respect to claims reciting features that are addressed in connection with the rejections applied to other claims pending in the appeal.

A. Claim 29 is directed to statutory subject matter, and the Examiner's rejection under 35 U.S.C. §101 should be reversed.

Claim 29 recites, in part “[a] program product, comprising . . . program code configured upon execution to . . . and; a recordable computer readable medium storing the program code.” Applicant specifically defines a “recordable medium” in the specification, at page 9, lines 16-20. This definition makes it clear that a “recordable computer readable medium” includes only tangible or physical type media, and specifically excludes signal-type media such as digital or analog communication links. The relevant passage in the specification, which defines both a “computer readable medium” and a “recordable medium,” is reproduced below, with additional paragraph breaks inserted to better illustrate that a “computer readable medium” includes recordable type media and transmission type media, but that recordable type media do not include transmission type media:

Examples of computer readable signal bearing media include but are not limited to

recordable type media such as volatile and non-volatile memory devices, floppy and other removable disks, hard disk drives, magnetic tape, optical disks (e.g., CD-ROMs, DVDs, etc.), among others, and
transmission type media such as digital and analog communication links. (Application, page 9, lines 16-20).

A medium that is “recordable” as defined in the specification includes only tangible and physical media (volatile and non-volatile memory devices, floppy and other removable disks, hard disk drives, magnetic tape, optical disks), and by virtue of the separate definition of “transmission type media,” specifically excludes the types of media that the Examiner is alleging to be non-statutory.

The Examiner argues in the final Office Action that claim 29 is not statutory because the claim includes program code. Claim 29, however, also includes a recordable computer readable medium in addition to program code, and the question of statutory subject matter rests on whether the claim as a whole recites statutory subject matter. Claim 16, for example, also includes program code, but the Examiner correctly did not reject that claim for lacking statutory subject matter because the claim recited other subject matter that was clearly statutory irrespective of whether program code is or is not statutory by itself.

A “recordable computer readable medium;” however, is statutory in nature. As noted above, a “recordable type media” is limited to tangible or physical media, and the term “recordable” is used to describe the claimed “computer readable medium.” The Examiner argues on page 3 of the Final Office Action, in addressing the aforementioned quoted passage, that “[t]he statement cited above not only does not distinguish the “readable medium” from the “transmission medium, but also clearly includes the “transmission medium” as a form of “readable medium.” Were claim 29 to merely recite a “readable medium,” the Examiner’s statement would be relevant. However, claim 29 quite clearly modifies the term “computer readable medium” with the term “recordable,” so the Examiner’s argument does not address Applicant’s claimed language.

The Examiner also argues, in the Advisory Action dated January 22, 2008, that the claim is “directed” to program code, and suggests that the claim would be statutory if directed to the medium that stores the program code. Applicant notes, however, that claim 29 is directed to a “program product” that comprises a program code and a medium, and is not merely directed to program code independent of any media. The Examiner’s position also seems to be far outside of the Office’s position with regard to “program product” claims, since a review of the patent database identifies over 18,000 issued United States patents that recite a “program product” in some form.

Applicant therefore respectfully submits that claim 29 is directed to statutory subject matter. Reversal of the Examiner’s §101 rejection is therefore respectfully requested.

B. Claims 1-11, 13-24 and 26-29 are patentable over Chan, and the Examiner's rejection under 35 U.S.C. §103(a) should be reversed.

Applicant respectfully submits that the Examiner's §103(a) rejections of claims 1-11, 13-24 and 26-29 based upon Chan are not supported on the record, and should be reversed, given that the Examiner has failed to establish a *prima facie* case of obviousness as to any of these claims. A *prima facie* showing of obviousness requires that the Examiner establish that the differences between a claimed invention and the prior art "are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art." 35 U.S.C. §103(a). Such a showing requires that all claimed features be disclosed or suggested by the prior art. Such a showing also requires objective evidence of the suggestion, teaching or motivation to combine or modify prior art references, as "[c]ombining prior art references without evidence of such a suggestion, teaching or motivation simply takes the inventor's disclosure as a blueprint for piecing together the prior art to defeat patentability -- the essence of hindsight." *In re Dembiczak*, 50 USPQ2d 1614, 1617 (Fed. Cir. 1999).

The rejection of each pending claim will be addressed hereinafter, starting first with the independent claims, and followed by the dependent claims.

Independent Claim 1

Independent claim 1 generally recites a method of executing a query in a database management system. The claimed method comprises receiving an SQL statement from an application program coupled to the database management system; executing the SQL statement; encrypting the SQL statement to generate an encrypted representation of the SQL statement; and logging execution of the SQL statement in a database monitor by storing the encrypted representation of the SQL statement in an execution log managed by the database monitor. The claim additionally recites displaying the execution log, which includes retrieving the encrypted representation of the SQL statement from the execution log, decrypting the encrypted representation of the SQL statement to generate an unencrypted representation of the SQL statement, and displaying the unencrypted representation of the SQL statement.

In rejecting claim 1, the Examiner relies solely on Chan, and in particular, col. 1, lines 65-67, col. 2, lines 48-67, col. 3, lines 11-60, col. 4 lines 12-60 and Fig. 4. Chan, however, does

not disclose any database monitor or execution log, much less a database monitor or execution log that encrypts any information stored therein.

A database monitor is a well known program used in a database management system to log the execution details of the system in an execution log, and is often used to optimize the system and/or the queries processed by the system. As discussed, for example, at page 3 of the Application, it has been found that in some instances, some of the execution details logged in an execution log by a database monitor may contain confidential information that a party may not wish to have accessible by others. For example, some execution details such as the values passed to parameter markers or host variables during execution of a query may include confidential information such as employee social security numbers, salary information, etc. In addition, query statements such as SQL statements may be confidential to an application developer, and a developer may prefer to keep those statements from being viewed by customers or competitors. Conventional database monitors, however, typically log all execution details for an executed query execution in an execution log, without regard to the type of execution detail. Consequently, if database monitoring is enabled, the potential exists that confidential information could be logged in a database monitor execution log and accessed by unauthorized parties.

The invention recited in claim 1 addresses this problem in part by encrypting a SQL statement that is being processed by a database management system such that an encrypted representation of that SQL statement is logged in the database monitor. Subsequently, whenever the database monitor is accessed to display all or a portion of the execution log, the SQL statement must be decrypted prior to being displayed. As a result, only parties authorized to decrypt the SQL statement will be able to view an unencrypted representation of the SQL statement when the execution log is displayed. Unauthorized parties, e.g., parties not having the appropriate decryption key, will not be able to view the unencrypted representation of the SQL statement.

Chan lacks any disclosure of encrypting an SQL statement and storing the encrypted SQL statement in an execution log of a database monitor. In fact, neither the term “log” nor the term “monitor” is even found in the reference. The passages cited by the Examiner merely disclose encrypting SQL statements passed by clients to a database management system as a mechanism

for allowing only trusted clients to access certain types of SQL statements (e.g., untrusted clients may only be able to issue read-type SQL statements, but trusted clients may be permitted to issue write-type SQL statements to enable only trusted clients to modify the contents of the database, *see* Chan, col. 1, lines 14-24). The encryption occurs prior to sending a query to a server, and the encrypted query is decrypted prior to executing the query (*see* Chan, Fig. 4, blocks 252 and 258, col. 4, lines 14-40).

Chan does not even disclose any encryption mechanism in a database server. Rather, clients are required to encrypt SQL statements, while a decryption engine in the server is used to decrypt those statements. Even if a conventional monitor is used in the Chan system (though no monitor is actually disclosed), there is no disclosure of any encryption engine within the Chan server that could encrypt SQL statements before they were stored in an encryption log. Of note, the cited passages at col. 3, lines 35-60 and col. 4, lines 50-60 refer to encryption operations performed in clients rather than any server in which a database monitor could be implemented.

Claim 1 also recites the display of the execution log, along with the retrieval, decryption and display of an unencrypted representation of an SQL statement that has been encrypted and stored in the execution log. In rejecting this feature, the Examiner relies on Fig. 4 and col. 4, lines 12-37 of Chan. However, these passages merely disclose the decryption of a query in connection with executing the query, and not in association with displaying an execution log from a database monitor.

The Examiner discounts this aspect of claim 1, arguing that displaying the content of stored information, including logs, is well known in the art. The Examiner also references the situation where a system administrator debugging a system looks into logs to review system history. Notably, however, Applicant is not merely claiming the display of an execution log. Instead, claim 1 is directed to displaying an SQL statement in an unencrypted form, and in connection with the display of an execution log, after than SQL statement has been encrypted and stored in the execution log. The fact that execution logs can be displayed relatively simply by system administrators, as noted by the Examiner, in fact proves the value of Applicant's claimed method, since the ease in which execution logs can be displayed by system administrators illustrates the ease in which system administrators could view potentially confidential or proprietary SQL statements issued by an application. As Applicant noted above,

in many cases an application developer may have a substantial interest in maintaining the confidentiality of his or her proprietary SQL statements. A developer's application, moreover, may issue queries to a database management system that is not under the control of the application developer, and as such, a system administrator for the database management system (who is typically not the application developer) conventionally can view the developer's SQL statements in unencrypted form in an execution log, which effectively destroys the confidentiality of the proprietary SQL statements. By encrypting the SQL statements in the execution log, and then decrypting such SQL statements in association with displaying the log, access to the SQL statements may be limited to those possessing the proper authorization, thus enabling a developer to maintain the confidentiality of his or her SQL proprietary work product.

In rebutting Applicant's arguments, starting at paragraph 5 of the Final Office Action, the Examiner first apparently argues that because Applicant's claimed database monitor is not a conventional database monitor, the concepts of a "monitor" and "log" need not be met by a database monitor or an execution log, and thus, Chan's "modified DBMS access procedure" corresponds to a "monitor" while stored items 242 correspond to a "log."

With all due respect, the Examiner's parsing of the claim language in this manner runs completely counter to the plain language of claim 1, and improperly reads limitations of the claim. Claim 1 does not merely recite a "monitor" or a "log," but specifically recites a "database monitor" and an "execution log," so ignoring the explicitly recited terms "database" and "execution" renders the rejection deficient on its face. A database monitor and an execution log are both expressly described in the specification, e.g., at pages 2-3 thereof, so the Examiner is bound to interpret these terms in a manner consistent with Applicant's specification. Given these descriptions, one of ordinary skill in the art would readily appreciate that a DBMS access procedure is not analogous to a database monitor, nor is a stored item 242 (Fig. 3 of Chan) analogous to an execution log. The DBMS access procedures disclosed in Chan are used to transmit queries to a DBMS for execution, and are not at all relevant to the concept of logging execution details about the execution of a query, as is the concern of a database monitor. Likewise, the stored items 242 are described at col. 3, lines 52-60 of Chan as constant strings that are stored in a procedure to identify the permissible SQL statements that a client is authorized to issue, and are not at all relevant to an execution log that stores the execution details regarding the

execution of a query. The DBMS access procedures and stored items, in fact, are used to initially generate an SQL statement to a DBMS for execution, but otherwise do not have any relevance to the actual execution of such statements, or more importantly, to logging the results of such execution.

Applicant also wishes to address the Examiner's novel argument that apparently holds that if a claimed element differs at all from a conventional element, the features of that conventional element need not be considered when interpreting the claimed element. The Examiner apparently takes the position that because Applicant claims a database monitor with encryption and decryption functionality, the claimed database monitor need not be interpreted as having the functionality of a conventional database monitor, so any element in a reference that can "monitor" something meets the claimed database monitor. Claim terms are interpreted in light of the specification, and Applicant has quite clearly described the characteristics of a database monitor and an execution log. That the claimed database monitor and execution log include additional functionality that encrypts and decrypts SQL statements stored in the execution log does not change the fact that the claimed database monitor is still a database monitor, and the claimed execution log is still an execution log, albeit with additional functionality. Applicant therefore respectfully submits that the DBMS access procedures and stored items relied upon by the Examiner are insufficient to establish a *prima facie* case of obviousness as to claim 1.

The Examiner also argues starting at page 4 of the Final Office Action that because displaying log files is conventional, and encrypting logged information is conventional, it follows necessarily that it is conventional to encrypt SQL statements in a database monitor execution log. The Examiner cites U.S. Patent No. 6,938,015 to illustrate the conventionality of encrypting logged information, but this reference merely discloses logging billing transactions, and is completely silent with respect to encrypting or logging execution details such as SQL statements in the execution log of a database monitor. Applicant submits that if it were so obvious to encrypt the execution log of a database monitor, as the Examiner asserts, the Examiner would be able to find a reference directed to such a feature. That the Examiner has to rely on the billing log of a consumer set top box illustrates that the concept of encrypting

execution details in a database monitor execution log is itself indicative of the non-obviousness of Applicant's claimed concept.

Applicant respectfully submits that Chan does not disclose either the encryption of SQL statements that are stored in an execution log of a database monitor, or the decryption of such statements in association with displaying an execution log. Moreover, given the unique and unexpected advantage of protecting potentially confidential execution details from being accessed via a database monitor, Applicant submits that claim 1 is non-obvious over Chan. The Examiner, furthermore, has provided no credible evidence of the recognition in the art of an objective reason to modify Chan to include Applicant's claimed database monitor execution log in which SQL statements are encrypted, stored, and then decrypted in association with displaying the execution log. Applicant therefore respectfully requests that the Examiner's rejection of claim 1 be reversed, and that claim 1, as well as claim 2 which depends therefrom, be passed to allowance.

Independent Claim 3

Next, with regard to the rejection of independent claim 3, this claim generally recites a method of logging query execution in a database management system, and recites generating an encrypted representation of an execution detail for a query executed by the database management system, logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log, and decrypting the execution detail in association with displaying the execution log.

In rejecting claim 3, the Examiner cites col. 3, lines 10-60 and col. 4, lines 50-60 of Chan. As noted above in connection with claim 1, however, Chan does not disclose or suggest the encryption of SQL statements or other execution details and the storage of same in an encrypted representation in an execution log. Chan also does not disclose or suggest decrypting an execution detail in association with displaying an execution log.

The cited passages in Chan, in particular, merely disclose functionality disposed in a client that encrypts a SQL statement prior to transmitting that statement to a database management system. Neither passage discloses or suggests encrypting an execution detail and storing that execution detail in an execution log in an encrypted representation.

In addition, the cited passages do not disclose decrypting an execution detail in association with displaying an execution log. The cited passages only refer to the decryption of an SQL statement by a database management system in connection with executing the statement. Again, no discussion is provided of a database monitor or execution log, and certainly no suggestion is provided for decrypting an SQL statement in connection with displaying any execution log. Furthermore, as discussed above in connection with claim 1, the Examiner cannot disregard the clear description in the specification defining the claimed “execution log.” The stored items displayed in Fig. 3 of Chan have nothing to do with logging execution details associated with the execution of a query. In addition, the fact that the display of an execution log may be known falls far short of disclosing or suggesting to one of ordinary skill in the art the concept of decrypting an execution detail in association with displaying an execution log.

Applicant therefore respectfully submits that the Examiner has failed to establish a *prima facie* case of obviousness as to claim 3. Reversal of the Examiner’s rejection of claim 3, and allowance of that claim and of claims 4-11 and 13-15 which depend therefrom, are therefore respectfully requested.

Independent Claim 16

Next, with respect to independent claim 16, this claim recites, similar to claim 3, the concept of generating an encrypted representation of an execution detail for a query executed by the database management system, logging the execution detail for the query in the execution log by storing the encrypted representation thereof in the execution log, and decrypting the execution detail in association with displaying the execution log. As discussed above in connection with claims 1 and 3, Chan does not disclose or suggest the concept of encrypting execution details, be they SQL statements or otherwise, within an execution log, as an execution log is described in the specification, nor does the reference disclose or suggest decrypting execution details in an execution log in association with displaying the execution log. Accordingly, claim 16 is non-obvious over Chan, and the rejection should be reversed. Reversal of the Examiner’s rejection of claim 16, and allowance of that claim and of claims 17-24 and 26-28 which depend therefrom, are therefore respectfully requested.

Independent Claim 29

Next, with respect to independent claim 29, this claim recites, similar to claim 3, the concept of generating an encrypted representation of an execution detail for a query executed by the database management system and logging the execution detail for the query in the execution log by storing the encrypted representation thereof in the execution log. As discussed above in connection with claims 1 and 3, Chan does not disclose or suggest the concept of encrypting execution details, be they SQL statements or otherwise, within an execution log, as an execution log is described in the specification. Accordingly, claim 29 is non-obvious over Chan. Reversal of the Examiner's rejection of claim 29, and allowance of the claim are respectfully requested.

Dependent Claims 2, 8, 9, 21 and 22

Claim 2 additionally recites encrypting at least one value passed to one of a host variable and a parameter marker used by the SQL statement, and that logging execution of the SQL statement includes storing the encrypted value in the execution log. Similarly, claims 8 and 21 recite that the execution detail comprises a value passed to a host variable during execution of the query, while claims 9 and 22 recite that the execution detail comprises a value passed to a parameter marker during execution of the query.

Host variables and parameter markers are described in the specification at page 3, lines 1-7. A host variable is defined as a variable defined in a host language, or that is generated by an SQL precompiler, and represents data to be inserted into the query during execution. A parameter marker, which is similar in function to a host variable, is defined as being capable of being used in dynamic SQL statements to represent positions in an SQL statement where data is to be inserted by an application.

In rejecting these claims, the Examiner relies on col. 3, lines 12-60 of Chan, arguing that the SQL statement, which is encrypted and sent during execution, corresponds to a value for a host variable. However, it is important to note that each of claims 2, 8, 9, 21 and 22 recites a value that is either "used by the SQL statement," (claim 2) or "passed . . . during execution of the query." The Examiner's use of the SQL statements themselves as corresponding to "values" effectively reads the fact that values are provided in addition to SQL statements or queries, out of these respective claims. Furthermore, as disclosed at col. 3, lines 50-60, the values that are

passed as arguments for placeholders in Chan are unencrypted – only the SQL statement string is encrypted. Thus, Applicant submits that even to the extent that Chan discloses encrypting an SQL statement, the encrypted SQL statement does not correspond to the values claimed in claims 2, 8, 9, 21 and 22. Furthermore, given that the SQL statement in Chan is not even stored in encrypted form in an execution log, or decrypted in association with the display of the execution log, as discussed above in connection with claims 1, 3 and 16, Applicant submits that Chan does not read on these claims. Reversal of the Examiner's rejections of claims 2, 8, 9, 21 and 22 are therefore respectfully requested.

Dependent Claims 4 and 17

Claims 4 and 17 are not argued separately.

Dependent Claims 5 and 18

Claims 5 and 18 additionally recite generating the encrypted representation after communicating the query to the database management system. The Examiner admits that this concept is not disclosed by Chan, but takes Official Notice that the concept is well known in the art. The Examiner asserts that a query is often announced to a DBMS as a preparatory handshake signal to determine if a server is active, and once the handshake is complete, the encryption proceeds.

Applicant first submits that the Examiner should be required to provide documentary evidence to support the taking of Official Notice, with a reference that illustrates this alleged conventional process. Applicant submits that the process that the Examiner alleges is common knowledge is not so instant and unquestionably demonstrable as being well-known in the art. *In re Ahlert*, 424 F.2d 1088, 1091, 165 USPQ 418, 420 (CCPA 1970), (citing *In re Knapp Monarch Co.*, 296 F.2d 230, 132 USPQ 6 (CCPA 1961)).

Second, the Examiner's argument, on its merits, is immaterial, because in the Examiner's scenario, Applicant assumes that even after the handshake occurs, the query would still be encrypted before the query itself is communicated over the network, otherwise, the purported purpose of encrypting the query in the first place would appear to be thwarted. The Examiner

does not allege that the query itself is transferred to the server during the handshake process, only that a handshake occurs.

Claims 5 and 18 are not directed to merely generating the encrypted representation after *contacting* a server. The claims require generating the encrypted representation after the query itself has been communicated to the DBMS, and the Examiner's Official Notice does not appear to take this scenario into account. Applicant therefore respectfully submits that no *prima facie* case of obviousness exists with respect to claims 5 and 28, so reversal of the Examiner's rejections of these claims are respectfully requested.

Dependent Claims 6-7 and 19-20

Claims 6-7 and 19-20 are not argued separately.

Dependent Claims 10 and 23

Claims 10 and 23 additionally recite logging a second execution detail for a query in the execution log in an unencrypted representation. The Examiner's rejection, however, merely discusses the fact that Chan logs statements in an encrypted form in a table, and is completely silent with respect to the concept of logging unencrypted execution details. The rejection is therefore deficient on its face.

In addition, to the extent that the Examiner considers Chan to disclose logging a query in an encrypted form in a "modified DBMS access procedure" as shown in Fig. 3, there is no disclosure in the reference that any other details associated with that query are similarly stored in the access procedure. The parameter values disclosed at col. 3 lines 50-60 do not appear to be stored or logged in the access procedure itself, so irrespective of the fact that the Examiner's analogizing of the access procedure to a database monitor or execution log constitutes faulty logic, these parameters are not "logged" in an unencrypted format even using the Examiner's tortured reading of the reference.

Applicant therefore respectfully submits that no *prima facie* case of obviousness exists with respect to claims 10 and 23, so reversal of the Examiner's rejections of these claims are respectfully requested.

Dependent Claims 11 and 24

Claims 11 and 24 depend respectively from claims 10 and 23, and additionally recite that the second execution detail (which is logged in unencrypted form) includes at least one of an access plan and a performance statistic associated with execution of the query. Thus, more so than any of the other claims at issue, these claims quite clearly address execution details, logged in unencrypted form, which are related to the execution and performance of a database query.

As is well known in the art, an access plan represents the executable representation of a database query, i.e., the optimized “plan of attack” for executing the query. Likewise, a performance statistic represents how the access plan performed when executing the query. As disclosed at page 6, lines 13-17, an execution details associated with an access plan may include, for example, join orders, and what indexes and other available resources were used, while performance statistics may include, for example, the time and amount of resources used to execute queries and the number of records returned as a result of queries.

In rejecting claims 11 and 24, the Examiner relies on col. 2, lines 60-67 and col. 3, line 60 to col. 4, line 5 of Chan. The Examiner argues that the second modified access program disclosed in these passages corresponds to an access plan. Considering, however, that the Examiner has analogized an access program to a database monitor, Applicant fails to appreciate how an access program, defined in this manner, can also represent an access plan for a database query. An access plan is well known in the art to be the output of an optimizer, and is the executable code for a database query (*see, e.g.*, page 10, lines 14-17 of the Application). Chan does not disclose logging any such information in an execution log. In fact, considering the Examiner argues that stored items 242 (which are defined within an access procedure 202, 206 of Chan, Fig. 3) correspond to an execution log, arguing that an access procedure 202 or 206 is an execution detail logged in an execution log would require the access procedure 202 or 206 to be stored within a stored item 242 – precisely the opposite configuration to that shown in Fig. 3. This non-sensical result occurs due to the tortured logic required to read Chan in such a way as to render obvious each and every feature of these claims and their respective independent claims. Applicant submits that, were Chan applied consistently to each and every feature of claims 11 and 24, no *prima facie* case of obviousness would be found. Applicant therefore respectfully

submits that no *prima facie* case of obviousness exists with respect to claims 11 and 24, so reversal of the Examiner's rejections of these claims are respectfully requested.

Dependent Claims 13-15 and 26-28

Claims 13-15 and 26-28 are not argued separately.

CONCLUSION

Applicant respectfully requests that the Board reverse the Examiner's rejections of claims 1-11, 13-24 and 26-29, and that the Application be passed to issue. If there are any questions regarding the foregoing, please contact the undersigned at 513/241-2324. If any other charges or credits are necessary to complete this communication, please apply them to Deposit Account 23-3000.

Respectfully submitted,

February 6, 2008
Date

/Scott A. Stinebruner/
Scott A. Stinebruner
Reg. No. 38,323
WOOD, HERRON & EVANS, L.L.P.
2700 Carew Tower
441 Vine Street
Cincinnati, Ohio 45202
Telephone: (513) 241-2324
Facsimile: (513) 241-6234

VIII. CLAIMS APPENDIX: CLAIMS ON APPEAL (S/N 10/671,343)

Listing of Claims:

1. (Previously Presented) A method of executing a query in a database management system, the method comprising:
 - receiving an SQL statement from an application program coupled to the database management system;
 - executing the SQL statement;
 - encrypting the SQL statement to generate an encrypted representation of the SQL statement;
 - logging execution of the SQL statement in a database monitor by storing the encrypted representation of the SQL statement in an execution log managed by the database monitor;
 - displaying the execution log, including retrieving the encrypted representation of the SQL statement from the execution log, decrypting the encrypted representation of the SQL statement to generate an unencrypted representation of the SQL statement, and displaying the unencrypted representation of the SQL statement.
2. (Original) The method of claim 1, further comprising encrypting at least one value passed to one of a host variable and a parameter marker used by the SQL statement, wherein logging execution of the SQL statement further comprises storing the encrypted value in the execution log.
3. (Previously Presented) A method of logging query execution in a database management system, the method comprising:
 - generating an encrypted representation of an execution detail for a query executed by the database management system;
 - logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log;
 - and
 - decrypting the execution detail in association with displaying the execution log.

4. (Original) The method of claim 3, further comprising receiving the query in an unencrypted form from an application program in communication with the database management system.

5. (Original) The method of claim 4, wherein generating the encrypted representation is performed after communicating the query to the database management system.

6. (Original) The method of claim 3, wherein generating the encrypted representation is performed prior to communicating the query to the database management system.

7. (Original) The method of claim 3, wherein the execution detail comprises a query statement.

8. (Original) The method of claim 3, wherein the execution detail comprises a value passed to a host variable during execution of the query.

9. (Original) The method of claim 3, wherein the execution detail comprises a value passed to a parameter marker during execution of the query.

10. (Original) The method of claim 3, further comprising logging a second execution detail for the query in the execution log in an unencrypted representation.

11. (Original) The method of claim 10, wherein the second execution detail includes at least one of an access plan and a performance statistic associated with execution of the query.

12. (Canceled).

13. (Previously Presented) The method of claim 3, wherein generating the encrypted representation includes encrypting the execution detail using a public key, and wherein

decrypting the execution detail includes decrypting the execution detail using a private key paired with the public key.

14. (Original) The method of claim 3, further comprising determining if database monitoring is enabled in the database management system, wherein generating the encrypted representation is performed if it is determined that database monitoring is enabled.

15. (Original) The method of claim 3, wherein the query comprises an SQL statement.

16. (Previously Presented) An apparatus, comprising:

at least one processor;

a memory within which is stored an execution log; and

program code configured to be executed by the at least one processor to log query execution in a database management system by generating an encrypted representation of an execution detail for a query executed by the database management system, logging the execution detail for the query in the execution log by storing the encrypted representation thereof in the execution log, and decrypting the execution detail in association with displaying the execution log.

17. (Original) The apparatus of claim 16, wherein the program code is further configured to receive the query in an unencrypted form from an application program in communication with the database management system.

18. (Original) The apparatus of claim 17, wherein the program code is configured to generate the encrypted representation after communicating the query to the database management system.

19. (Original) The apparatus of claim 16, wherein the program code is configured to generate the encrypted representation prior to communicating the query to the database management system.

20. (Original) The apparatus of claim 16, wherein the execution detail comprises a query statement.

21. (Original) The apparatus of claim 16, wherein the execution detail comprises a value passed to a host variable during execution of the query.

22. (Original) The apparatus of claim 16, wherein the execution detail comprises a value passed to a parameter marker during execution of the query.

23. (Original) The apparatus of claim 16, wherein the program code is further configured to log a second execution detail for the query in the execution log in an unencrypted representation.

24. (Original) The apparatus of claim 23, wherein the second execution detail includes at least one of an access plan and a performance statistic associated with execution of the query.

25. (Canceled).

26. (Previously Presented) The apparatus of claim 16, wherein the program code is configured to generate the encrypted representation by encrypting the execution detail using a public key, and wherein the program code is configured to decrypt the execution detail by decrypting the execution detail using a private key paired with the public key.

27. (Original) The apparatus of claim 16, wherein the program code is further configured to determine if database monitoring is enabled in the database management system, and wherein the program code is configured to generate the encrypted representation if it is determined that database monitoring is enabled.

28. (Original) The apparatus of claim 16, wherein the query comprises an SQL statement.

29. (Previously Presented) A program product, comprising:

program code configured upon execution to log query execution in a database management system by generating an encrypted representation of an execution detail for a query executed by the database management system, and logging the execution detail for the query in an execution log for the database management system by storing the encrypted representation thereof in the execution log; and

a recordable computer readable medium storing the program code.

30. (Canceled).

IX. EVIDENCE APPENDIX

10/671,343

None.

X. RELATED PROCEEDINGS APPENDIX

10/671,343

None.